Terraform -   
- do you working experience provisioning production environment using Terraform?  
- name a few types of azure resources that you have provisioned using terraform?  
- where does your statefile lives?

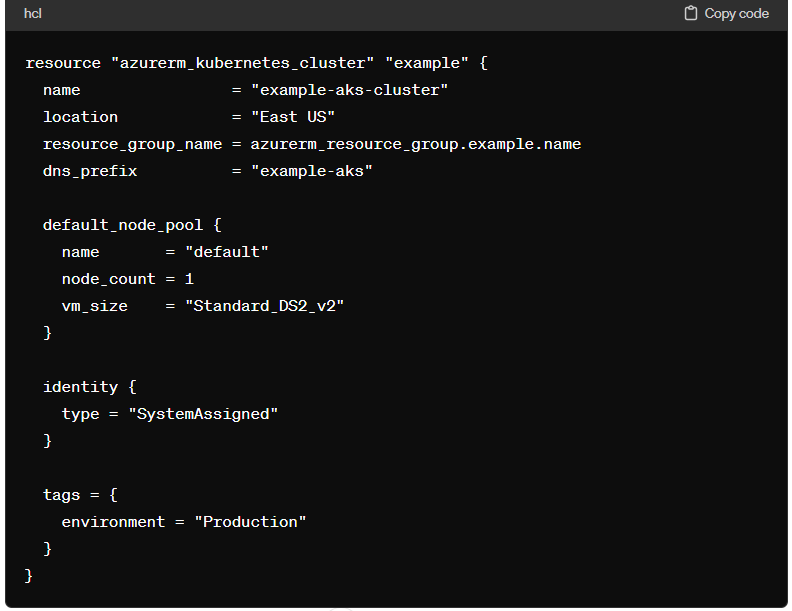
 Azure -   
- did you upgraded AKS for kubernetes version?  
- did you configured private endpoints for storage accounts or key vaults in azure?  
- have worked with azure cli and azure RM?

Kubernetes -

* have you used helm charts to deploy micro services?
* which Kubernetes client you use to debug micro services? (answer can be any of following: Lense, k9s, kubectl
* have you used Flux CD or similar tool to automate deployments?

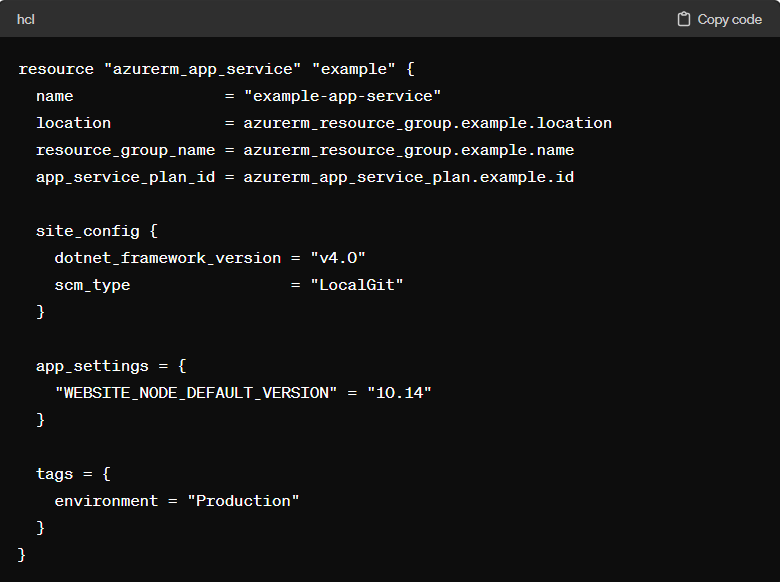
**Terraform:**

* **Q: Do you have working experience provisioning production environments using Terraform?**
  + A: Yes, I have extensive experience provisioning production environments using Terraform. For example, I've used Terraform to provision a production-grade Azure Kubernetes Service (AKS) cluster along with other necessary resources such as virtual networks, storage accounts, and Azure App Services. Here's a simplified example of how I provisioned an AKS cluster using Terraform:



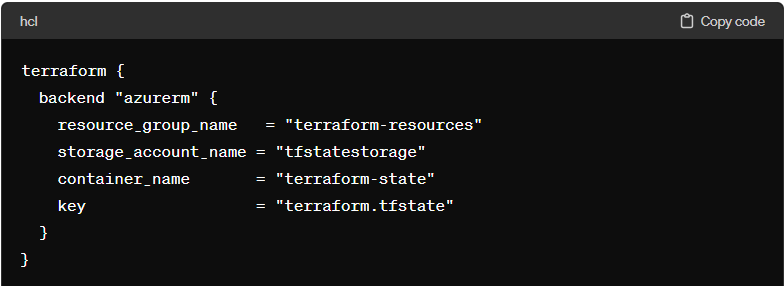
**Q: Name a few types of Azure resources that you have provisioned using Terraform?**

* A: I have provisioned various Azure resources using Terraform, including virtual networks, storage accounts, Azure Kubernetes Service (AKS) clusters, and Azure App Services. For example, here's how I provisioned an Azure App Service using Terraform:



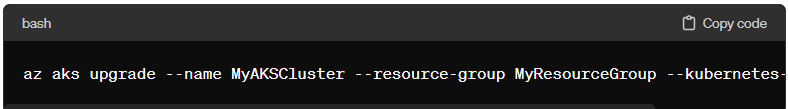
**Q: Where does your state file live?**

* A: I typically store the Terraform state file in an Azure Storage Account or using Azure Remote State. For example, I use Azure Blob Storage to store my Terraform state file. Here's how I configure Terraform to store state in Azure Blob Storage:



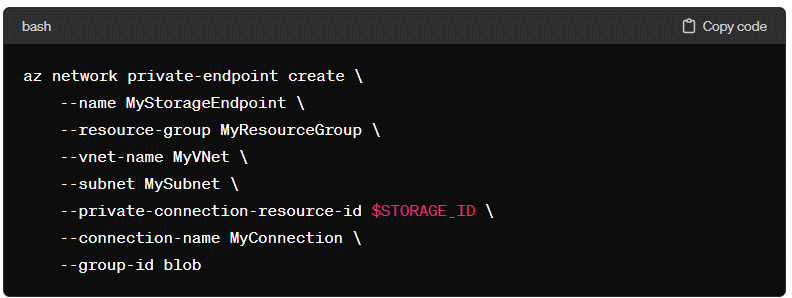
**Azure:**

* **Q: Did you upgrade AKS for Kubernetes version?**
  + A: Yes, I have upgraded AKS clusters to newer Kubernetes versions. For example, I recently upgraded an AKS cluster from Kubernetes version 1.18 to 1.19. Here's how I upgraded the AKS cluster using Azure CLI:



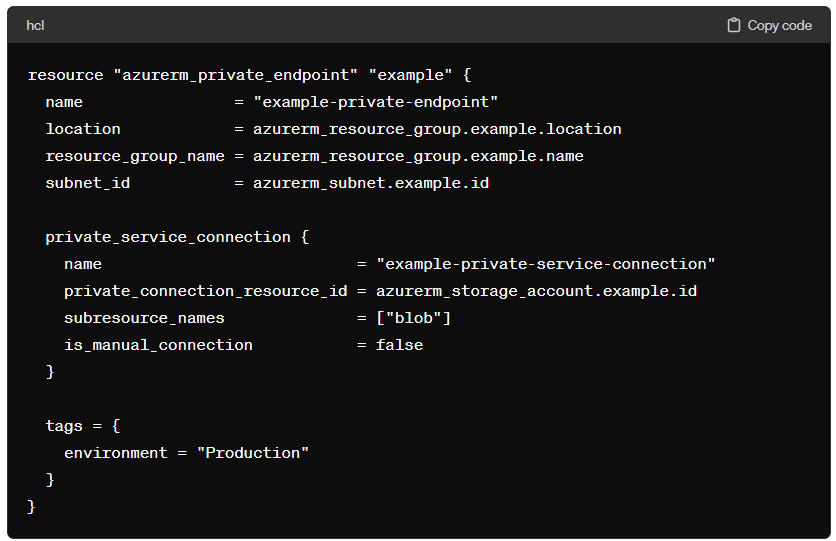
**Q: Did you configure private endpoints for storage accounts or Key Vaults in Azure?**

* A: Yes, I have configured private endpoints for both storage accounts and Key Vaults in Azure. For instance, I configured a private endpoint for a storage account to enable secure access from a virtual network. Here's how I configured a private endpoint for an Azure Storage Account using Azure CLI:



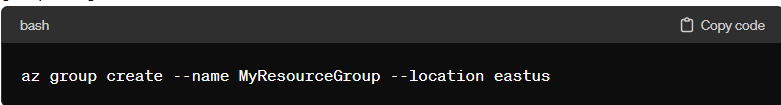
Or

A: Yes, I have configured private endpoints for both storage accounts and Key Vaults in Azure. For instance, I configured a private endpoint for a storage account to enable secure access from a virtual network. Here's how I configured a private endpoint for an Azure Storage Account using Terraform:



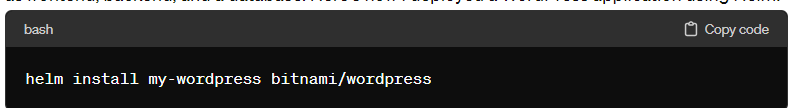
**Q: Have you worked with Azure CLI and Azure RM?**

* A: Yes, I have experience working with both Azure CLI and Azure Resource Manager (Azure RM). For example, I've used Azure CLI to automate the deployment of Azure resources and Azure RM templates to define infrastructure as code. Here's an example of how I create an Azure resource group using Azure CLI:



**Kubernetes:**

* **Q: Have you used Helm charts to deploy microservices?**
  + A: Yes, I have used Helm charts extensively to deploy microservices on Kubernetes. For example, I used Helm charts to deploy a microservice-based application consisting of multiple services, such as frontend, backend, and a database. Here's how I deployed a WordPress application using Helm:



A: Yes, I have used Helm charts extensively to deploy microservices on Kubernetes. Helm is a package manager for Kubernetes that helps in defining, installing, and upgrading even the most complex Kubernetes applications. Using Helm charts, I've deployed microservice-based applications consisting of multiple services, such as frontend, backend, and a database, with ease.

For example, let's say I wanted to deploy a WordPress application using Helm. I would first create a Helm chart for WordPress or use an existing one from a chart repository. Then, I would install the Helm chart using the **helm install** command:

bash

This command will deploy the WordPress application along with all its dependencies, such as a MySQL database, as defined in the Helm chart. Helm charts make it easy to manage and deploy complex applications on Kubernetes, providing a consistent and repeatable deployment process.

**Q: Which Kubernetes client do you use to debug microservices? (Answer can be any of the following: Lens, k9s, kubectl)**

* A: I primarily use kubectl to debug microservices running on Kubernetes clusters. For example, I use **kubectl logs** to view logs from containers, **kubectl exec** to execute commands within containers, and **kubectl describe** to get detailed information about Kubernetes resources.



**Q: Have you used Flux CD or a similar tool to automate deployments?**

* A: Yes, I have used Flux CD to automate deployments on Kubernetes clusters. For instance, I've configured Flux CD to automatically sync changes from a Git repository to a Kubernetes cluster, ensuring continuous deployment of applications. Here's how I installed Flux CD on my Kubernetes cluster:

